UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/753,992	01/03/2001	Sanjay Khanna	RSW919990130US1	1791

7590 04/20/2004

Jerry W. Herndon
IBM Corporation T81/503
PO Box 12195
Research Triangle Park, NC 27709

EXAMINER	
CHEN, CHONGSHAN	
ART UNIT	PAPER NUMBER
2172	13

DATE MAILED: 04/20/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary

Application No. 09/753,992	**Applicant(s)** KHANNA ET AL.	
Examiner Chongshan Chen	**Art Unit** 2172	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *05 February 2004*.

2a)☒ This action is **FINAL**.   2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *25,27 and 29-45* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *25, 27 and 29-45* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date _____.

4) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application (PTO-152)

6) ☐ Other: _____.

DETAILED ACTION

Response to Arguments

1. Applicant's arguments filed on 5 February 2004 have been fully considered but they are not persuasive.

2. As per applicant's arguments regarding Gorelik teaches using two copies of a single database, while the applicants' invention uses two indexes to a single copy of the data, rather than duplicating the underlying data have been considered but are not persuasive. First, there is no limitation "a single copy" in independent claims 25 and 27. Second, in other independent claims, the claim language, each data structures/indexes represents an initial state for accessing a single copy of stored data (or each indexing a single copy of the stored data), is different from the applicants' arguments, because each data structures/indexes represents a single copy of the data is different from both (or two) indexes represent a single copy of the data. In Gorelik's reference, there are two identical copies of database, then of course there are two identical data structures/indexes to the two identical databases. Each data structures/indexes does not point to both databases; each data structures/indexes only points to its corresponding single copy of the database. Therefore, Gorelik reads on the claim language. The argument is not persuasive.

3. As per applicant's arguments regarding Gorelik does not teach performing an update and then switching the databases responsive to the update and the examiner is using hindsight reconstruction against applicants' claims have been considered but are not persuasive. In response to applicant's argument that the examiner's conclusion of obviousness is based upon improper hindsight reasoning, it must be recognized that any judgment on obviousness is in a

sense necessarily a reconstruction based upon hindsight reasoning. But so long as it takes into account only knowledge which was within the level of ordinary skill at the time the claimed invention was made, and does not include knowledge gleaned only from the applicant's disclosure, such a reconstruction is proper. See *In re McLaughlin*, 443 F.2d 1392, 170 USPQ 209 (CCPA 1971). Gorelik teaches manually switching databases (Gorelik, page 3, [0036]). It would have been obvious to one of ordinary skill in the art at the time the invention was made to switch after each update because after each update, one points to newly updated content, the other points to old incorrect content. If do not switch after update, then query will search the old incorrect content and might return incorrect result. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to switch responsive to each update in order to allow the query to search the newly updated content and retrieve correct result.

4. As per applicant's arguments regarding the references neither teaches a search use count is incremented for each search and then decremented after the search nor teaches this search use count has a value indicating that no searches are being performed have been considered but are not persuasive. "Serialization of AVL-Binary Tree Element Retrieval via Duplexed Pointers" teaches a search use count is incremented for each search (page 1, "the synchronization count is incremented when an update is made to the AVLDP tree"). Clearly, if the search use count is not incremented, it indicates that no search is being performed. Furthermore, "Serialization of AVL-Binary Tree Element Retrieval via Duplexed Pointers" teaches the synchronization count and active tree indicator are compared with ones obtained at the start of the retrieve after completion. If they are the same, the retrieve was successful. Since the synchronization (search use) count is

incremented for each search, the synchronization (search use) count needs to be decremented after completion so that the synchronization count can be the same as the one obtained at the start of the retrieve to indicate the retrieve was successful. Therefore, "Serialization of AVL-Binary Tree Element Retrieval via Duplexed Pointers" teaches decreasing the search use count after the search. The arguments are not persuasive.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

7. Claims 25, 27 and 29-45 are rejected under 35 U.S.C. 103(a) as being unpatentable over Gorelik et al. ("Gorelik", Pub. No.: US 2002/0004799) in view of "Serialization of AVL-Binary Tree Element Retrieval via Duplexed Pointers" (March 1992, IBM technical disclosure bulletin, page 138-139).

As per claim 25, Gorelik teaches a method of searching and updating a database in a multi-processing environment, comprising the steps of:

maintaining two databases, first database for searching and a second database for updating (Gorelik, Fig. 1, Fig. 4A-4E, page 1, [0008]-[0009], "a system maintaining two copies of a database to be accessed by the system's application … processing the request by a first database, when the request is for a read operation, and processing the request by a second database, when said request is for write/load operation"),

responsive to each update of the second database, switching the databases so that the first database becomes the second database and the updated database becomes the first database (Gorelik, Fig. 1, Fig. 4A-4E, page 2, [0023], "DB A is the live database and fields queries from applications, while DB B is the load database and receives update … in due course, control manager 18 switches the designations … then DB A would be the load database and DB B would be the live database"),

allowing searches that are in progress using the first database, before the switching, to continue until completion after the switching, using what is then the second database (Gorelik, page 3, [0043], "A Reconcile Pending timeout might be used to allow a query to finish before reconciliation starts"),

after the switching, initiating new searches using what is then the first index (Gorelik, Fig. 4A-4E, page 3, [0040], "Once a database is loaded successfully, a switch can take place such that the user applications are redirected to the newly loaded database and that database becomes the new live database"), when all searches in what is then, after the switching, the second database have completed, updating what is then the second database in an identical manner as the update to which the switching step was responsive (Gorelik, page 3, [0043], "A reconciliation utility may move the update data to the live database ... A Reconcile Pending timeout might be used to allow a query to finish before reconciliation starts"); and

preventing another operation of the switching step until completion of the step of updating the second index in the identical manner (Gorelik, Fig. 4A-4E, page 3, [0040], "Once a database is loaded successfully, a switch can take place").

Gorelik uses two identical copies of a database. However, Gorelik does not explicitly disclose using indexes to the database. "Serialization of AVL-Binary Tree Element Retrieval via Duplexed Pointers" teaches using indexes to the database in searching and updating (first page of the article). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to use index to the database in search and update in the method of Gorelik because using indexes improves processing speed.

Claim 27 is rejected on grounds corresponding to the reasons given above for claim 25.

As per claim 29, Gorelik teaches a computer program product for serializing data structure retrievals and updates in a multi-processing computer system, the computer program product embodied on one or more computer-readable media and comprising:

computer-readable program code means for creating two identical data structures, each representing an initial state for accessing a single copy of stored data (Gorelik, Fig. 1, page 1, [0008]-[0010], two identical copies of databases of course have two identical data structures, each data structure does not point to the multiple copies, it only points to its corresponding single copy of data, which reads on the claim language of "each representing …");

computer-readable program code means for performing searches against a first of the two data structure (Gorelik, Fig. 1, Fig. 4A-4E, page 2, [0023]);

computer-readable program code means for performing a first update against a second of the two data structure, yielding a revised data structure (Gorelik, Fig. 1, Fig. 4A-4E, page 2, [0023]);

computer-readable program code means for switching the first data structure and the revised data structure, responsive to completion of the computer-readable program code means for performing the first update, such that the first data structure becomes the second data structure and the revised data structure becomes the first data structure, said the computer-readable program code means for switching the data structures further comprising a third instruction for re-ordering data structure pointers atomically to prevent interference from other processes during operation of the computer-readable program code means for switching (Gorelik, Fig. 4A-4E, page 2, [0023]); and

computer-readable program code means for applying, after operation of the computer-readable program code means for switching, the first update against the second data structure, yielding a second data structure that is structurally identical to the first data structure (Gorelik, Fig. 4A-4E, page 3, [0040]);

the computer-readable program code means for performing searches further comprising computer-readable program code means for activating the computer-readable program code means for applying the first update against the second data structure when the search use count for the second data structure has a value indicating that no search are being performed against the second data structure.

Gorelik does not explicitly teach a first program instruction for incrementing a search use count for the first data structure atomically during each search and a second instruction for decrementing the search use count for the first data structure atomically after performing each search; applying the first update against the second data structure when the search use count for the second data structure has a value indicating that no search are being performed against the second data structure. However, Gorelik teaches update the second database after all the searches are finished (Gorelik, page 3, [0043]). "Serialization of AVL-Binary Tree Element Retrieval via Duplexed Pointers" teaches a synchronization count, and each time an update is performed, the synchronization count is incremented (page 138, 2nd paragraph). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to include a use count in the system of Gorelik to track how many processors are accessing the database. Because the read and write operations cannot be applied to a database at the same time, the update operation can be applied to a database only when there are no read operation in the database. Therefore, the database management system would include the use count to track how many search operations are in the database, and update the database only when the use count is zero (no search in the database) in order to preserve the integrity of the database.

As per claim 30, Gorelik and "Serialization of AVL-Binary Tree ..." teach all the claimed subject matters as discussed in claim 29, and further teach computer-readable program code means for obtaining an exclusive lock on the second data structure prior to operation of the computer-readable program code means for performing the first update ("Serialization of AVL-

Binary Tree …", first page, last paragraph, lines 1-3); and computer-readable program code means for releasing the exclusive lock after operation of the computer-readable program code means for applying the first update ("Serialization of AVL-Binary Tree …", second page, first paragraph, lines 1-3).

As per claim 31, Gorelik and "Serialization of AVL-Binary Tree …" teach all the claimed subject matters as discussed in claim 29, and further teach computer-readable program code means for queuing a transaction that specifies one or more data structure traversals and one or more data structure modifications that were performed to yield the revised data structure, and wherein the computer-readable program code means for applying the first update further comprises computer-readable program code means for performing the one or more data structure traversals and the one or more modifications specified in the queued transaction against the second data structure that results from operation of the computer-readable program code means for switching (Gorelik, 4A-4E, page 3, [0040], "Serialization of AVL-Binary Tree …", page 1-2 of the article).

As per claim 32, Gorelik and "Serialization of AVL-Binary Tree …" teach all the claimed subject matters as discussed in claim 29, and further teach performing a subsequent update against the second data structure that results from operation of the computer-readable program code means for applying the first update; and wherein operation of the computer-readable program code means for performing the subsequent update causes another operation of the computer-readable program code means for switching and the computer-readable program code means for applying (Gorelik, Fig. 4A-4E).

Claims 33-36 are rejected on grounds corresponding to the reasons given above for claims 29-32.

Claims 37-40 are rejected on grounds corresponding to the reasons given above for claims 29-32.

As per claim 41, Gorelik teaches a method for serializing data retrievals and updates in a computing environment, comprising steps of:

creating two identical databases, each representing an initial state for accessing stored data (Gorelik, Fig. 1, page 1, [0008]-[0010]);

performing searches against a first of the two databases (Gorelik, Fig. 1, Fig. 4A-4E, page 2, [0023]);

performing a first update against a second of the two databases, yielding a revised database (Gorelik, Fig. 1, Fig. 4A-4E, page 2, [0023]);

serializing information on how the first update affected the second database, including how the second database was traversed for making the first update and how the second database was modified in the first update (Gorelik, Fig. 4A-4E, page 2, [0023], page 3, [0040]);

switching the first database and the revised database, responsive to performing the first update, such that the first database becomes the second database and the revised database becomes the first database (Gorelik, Fig. 4A-4E, page 2, [0023]);

applying, after the switching step, the serialized information to the second database, using the information about how the second database was traversed and modified to efficiently traverse and modify the newly-switched second database, thereby yielding a second database that is synchronized with, arid structurally identical to, the first database (Gorelik, Fig. 4A-4E, page 3, [0040]); and

performing subsequent searches against the first database (Gorelik, Fig. 4A-4E).

Gorelik uses two identical copies of a database. However, Gorelik does not explicitly disclose using indexes to the database. “Serialization of AVL-Binary Tree Element Retrieval via Duplexed Pointers” teaches using indexes to the database in searching and updating (first page of the article). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to use index to the database in search and update in the method of Gorelik because using indexes improves processing speed.

As per claim 42, Gorelik and “Serialization of AVL-Binary Tree …” teach all the claimed subject matters as discussed in claim 41, and further teach performing a subsequent update against the second index that results from applying the serialized information; and wherein the step of performing the subsequent update causes repeating the serializing, switching, and applying steps (Gorelik, Fig. 4A-4E).

Claim 43 is rejected on grounds corresponding to the reasons given above for claims 37-40.

As per claim 44, Gorelik and “Serialization of AVL-Binary Tree …” teach all the claimed subject matters as discussed in claim 41, and further teach the indexes are implemented as trees (“Serialization of AVL-Binary Tree …”, page 1).

As per claim 45, Gorelik and “Serialization of AVL-Binary Tree …” teach all the claimed subject matters as discussed in claim 41, except for explicitly disclosing the indexes are implemented as hash tables. However, “Serialization of AVL-Binary Tree …” teaches the indexes are implemented as trees ("Serialization of AVL-Binary Tree ...", page 1). It is well known to one of ordinary skill in the art that a tree can be implemented as a hash table in a program environment. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to implement the tree indexes of "Serialization of AVL-Binary Tree ..." as hash tables in a program environment since a programmer usually use hash tables to implement tree structure.

Conclusion

8. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Contact Information

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Chongshan Chen whose telephone number is 703-305-8319. The examiner can normally be reached on Monday - Friday (8:00 am - 4:30 pm).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, John E Breene can be reached on (703)305-9790. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

April 12, 2004

SHAHID ALAM
PRIMARY EXAMINER